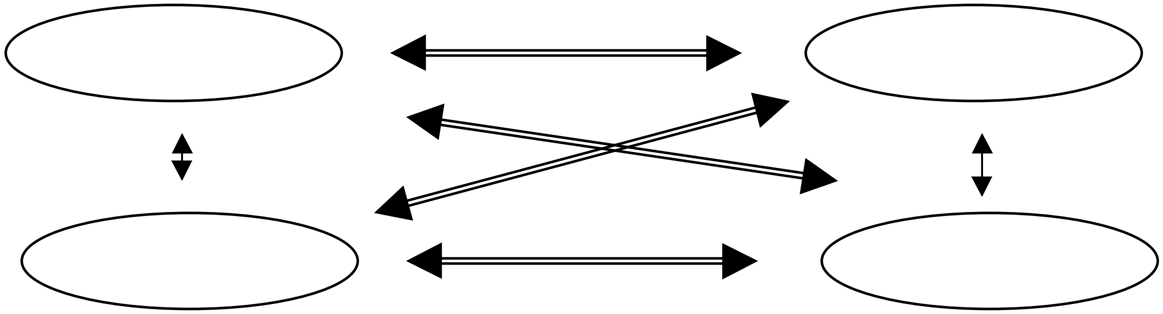
Path Development in a PKI Network Environment

Santosh Chokhani & Peter M. Hesse

chokhani@cygnacom.com

Path Development: Problem

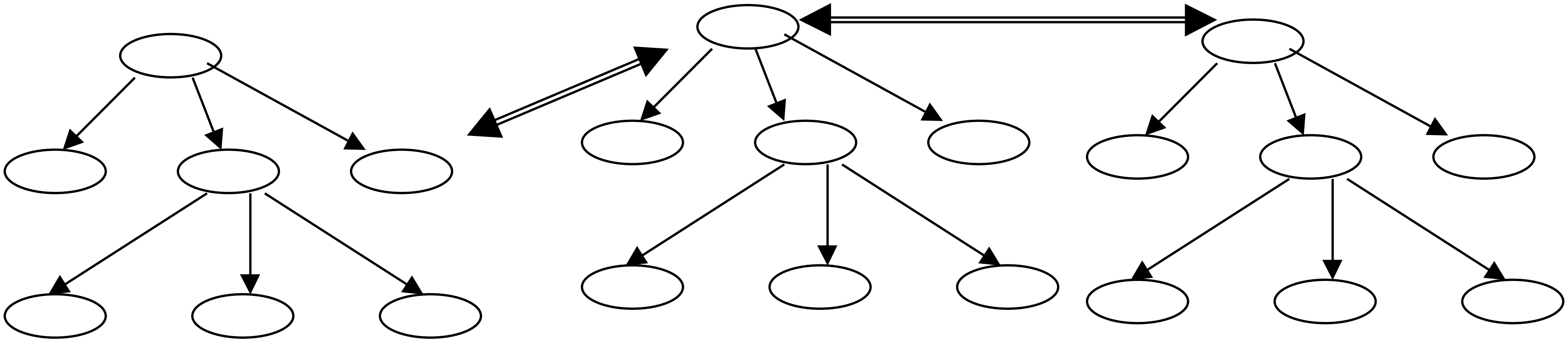
- **Discover Path in Network of CAs - Hard**



CYGNACOM SOLUTIONS

Path Development: Problem

- Discover path in cross certified hierarchy
  - easier



CYGNACOM SOLUTIONS

Path Development: CygnaCom Approach

- Assume network
- Give preference to caCertificate attribute over crossCertificatePair attribute
- Use version 3 extensions and matching rules: keyUsage = certSign, path to name, subjectKeyIdentifier, validity period, algorithmIdentifier

Path Development Procedure

- Set subject DN = end entity DN
- keyUsage = bit set for digital signature (if signature verification is required), or key encipherment or key agreement
- certificateValid = current date and time in ZULU
- subjectKeyIdentifier if known (e.g., from application protocol)
- pathToName equal to the end entity name
- attribute = userCertificate
- obtain the certificate from directory. If no certificate is available, then backtrack.

Path Development Procedure (continued)

- If the issuer DN in the certificate equals a trusted CA and signature on the certificate verifying using a trusted CA public key, then stop; path development is complete.
- Else, subject DN = issuer DN in certificate
- attribute = caCertificate & crossCertificatePair
- subjectKeyIdentifier = issuer key identifier in the certificate (authorityKeyIdentifier)
- keyUsage = bit set for certificate signature (certSign)
- Go to step to obtain certificate from directory.

Path Development Procedure (concluded)

- Use backtracking
- Use graph traversal algorithm for determining when a node is revisited (DN and certificates)
- Helps detect loops
- Prioritize certificates
- Will have digressions, but matching rules help reduce them
- Algorithm efficient for networks

Path Development

Certificate Graph

Graph Node Data Structure

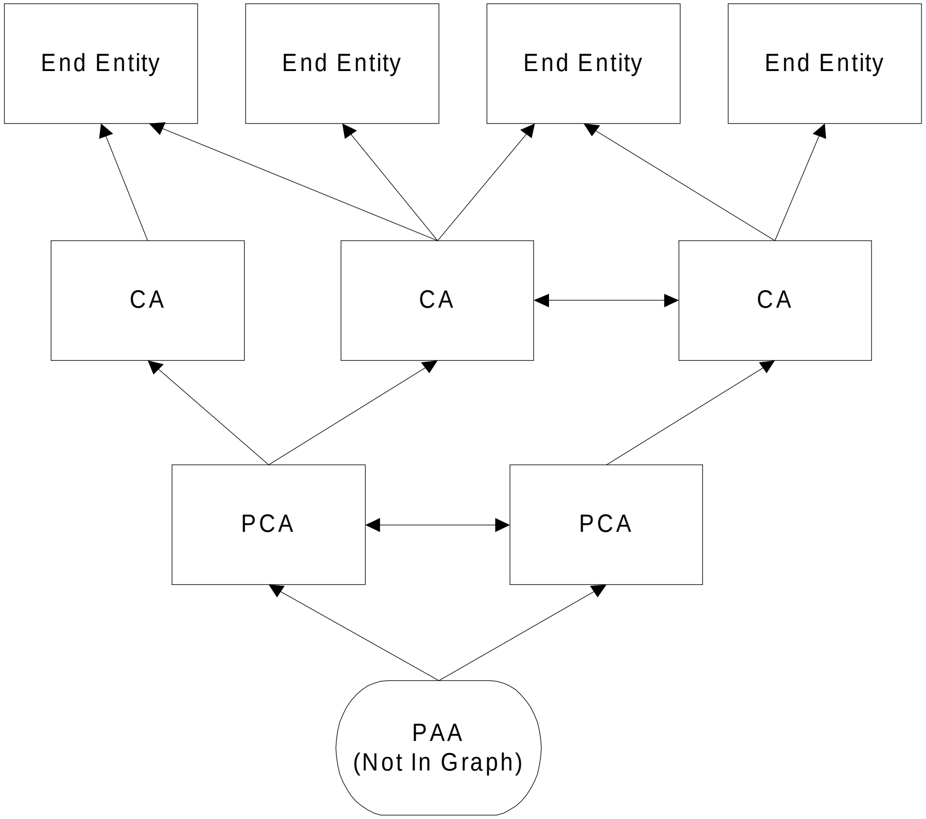
Path Development Algorithm

Certificate List Prioritization

Certificate Retrieval

Certificate Graph

- This is a graphical representation of the Certificate Graph Data Structure. Each node represents a different distinguished name.
- This graph is built and traversed during path development.
- The path is developed from the top to the bottom of this graph

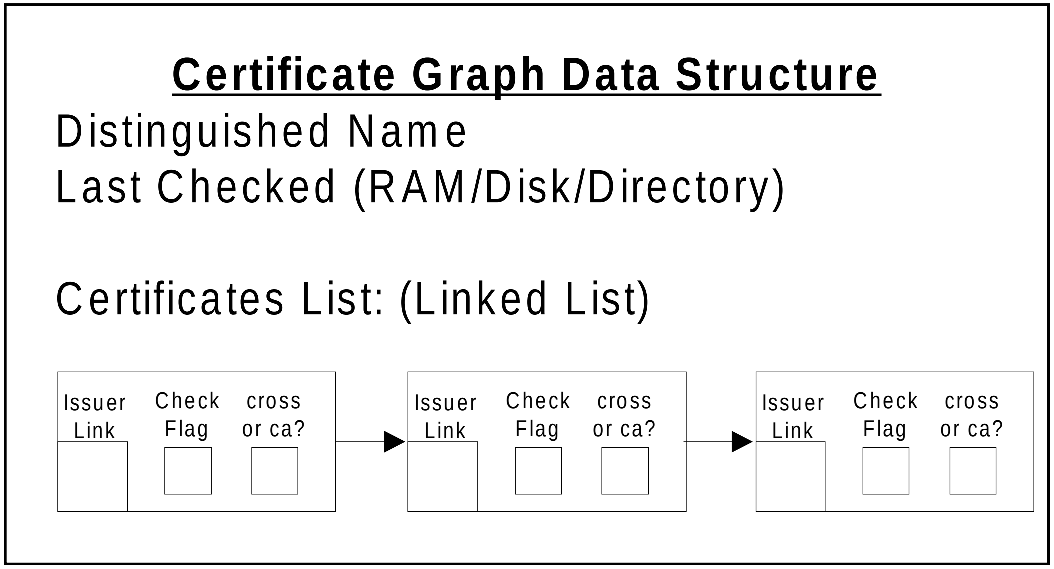


Graph Node Data Structure

This is the contents of one node from the Certificate Graph Data Structure.

Last Checked: Last location that was checked for certificates for this DN

Certificates List: This is a linked list of all certificates for this DN. This list may be incomplete if Last Checked is not the Directory.



Certificate Graph Data Structure

Distinguished Name

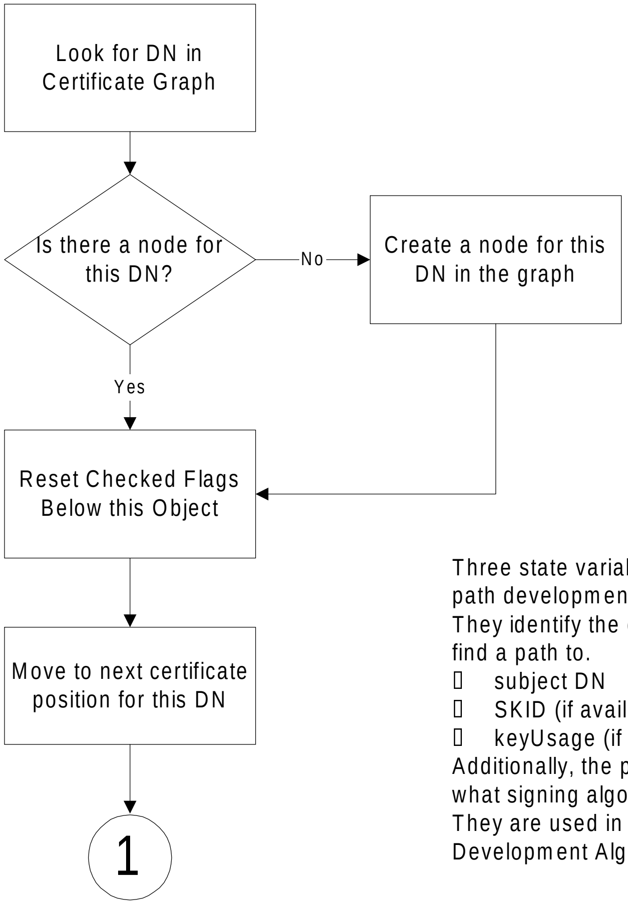
Last Checked (RAM/Disk/Directory)

Certificates List: (Linked List)

Certificate list items:

- Issuer link saves a link to the issuer of a certificate (once determined)
- Check flag is TRUE if this certificate has been tried in the current path development
- Cross or CA saves if this was a caCertificate or from crossCertificatePair

Path Development Algorithm (1)

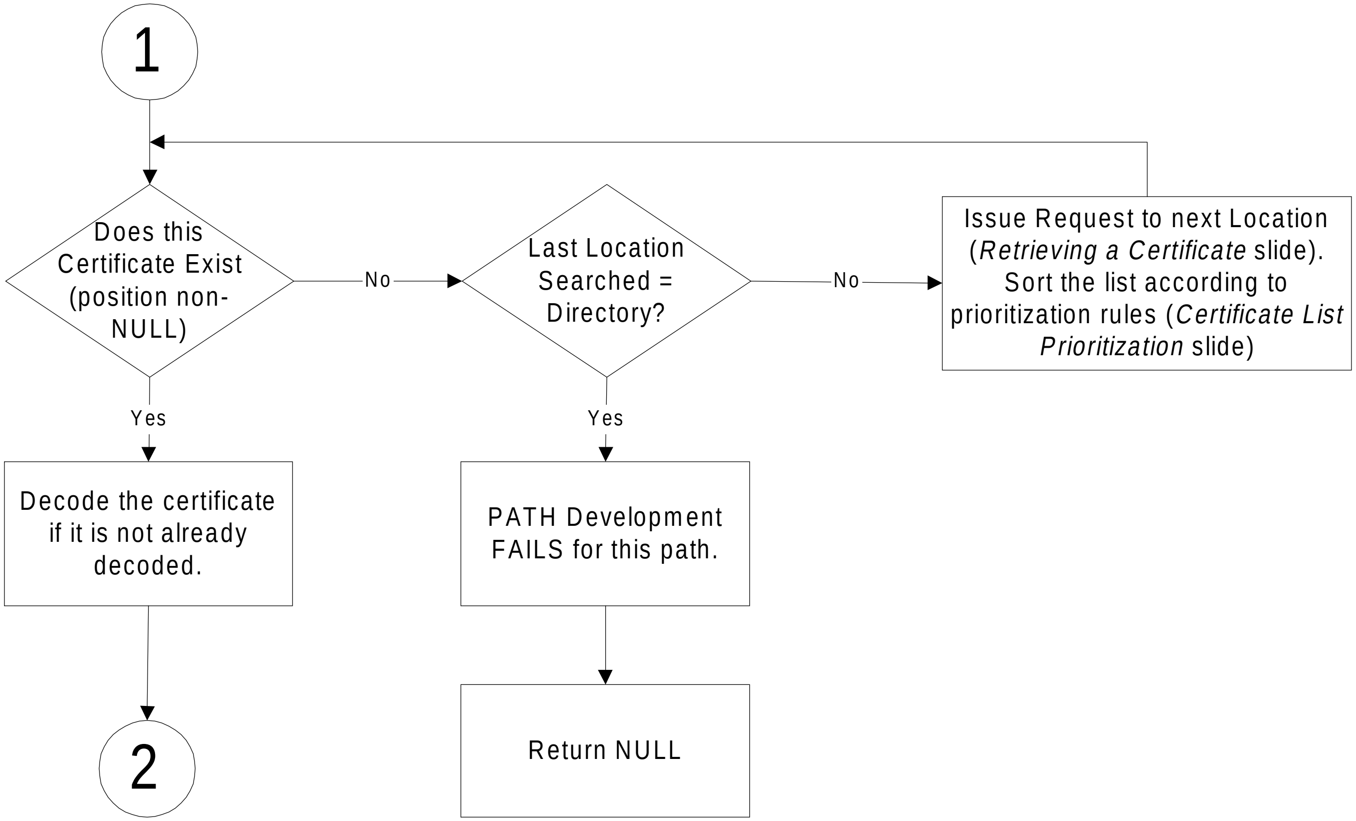

Look for DN in Certificate Graph
Is there a node for this DN?
No
Create a node for this DN in the graph
Yes
Reset Checked Flags Below this Object
Move to next certificate position for this DN
1

Three state variables are passed into the path development algorithm at the start. They identify the certificate you wish to find a path to.

- subject DN
- SKID (if available)
- keyUsage (if available)

Additionally, the process is made aware of what signing algorithms are acceptable. They are used in the third Path Development Algorithm slide

Path Development Algorithm (2)

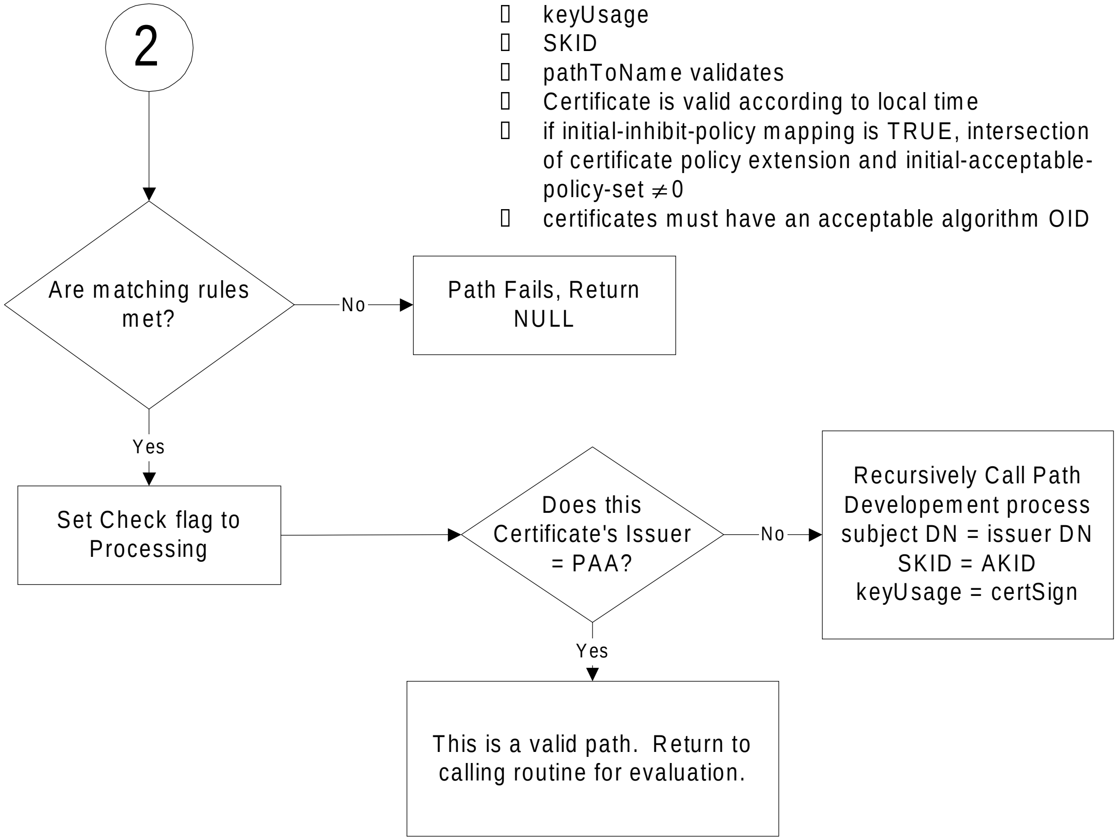

1
Does this Certificate Exist (position non-NULL)
No
Last Location Searched = Directory?
No
Issue Request to next Location (*Retrieving a Certificate* slide). Sort the list according to prioritization rules (*Certificate List Prioritization* slide)
Yes
Yes
Decode the certificate if it is not already decoded.
PATH Development FAILS for this path.
2
Return NULL

CYGNACOM SOLUTIONS

Path Development Algorithm (3)

Matching rules:

- keyUsage
- SKID
- pathToName validates
- Certificate is valid according to local time
- if initial-inhibit-policy mapping is TRUE, intersection of certificate policy extension and initial-acceptable-policy-set $\neq 0$
- certificates must have an acceptable algorithm OID



2

Are matching rules met?

No

Path Fails, Return NULL

Yes

Set Check flag to Processing

Does this Certificate's Issuer = PAA?

No

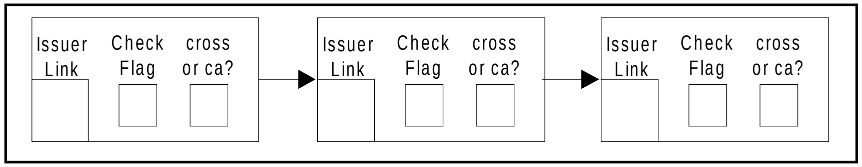
Recursively Call Path Developement process
subject DN = issuer DN
SKID = AKID
keyUsage = certSign

Yes

This is a valid path. Return to calling routine for evaluation.

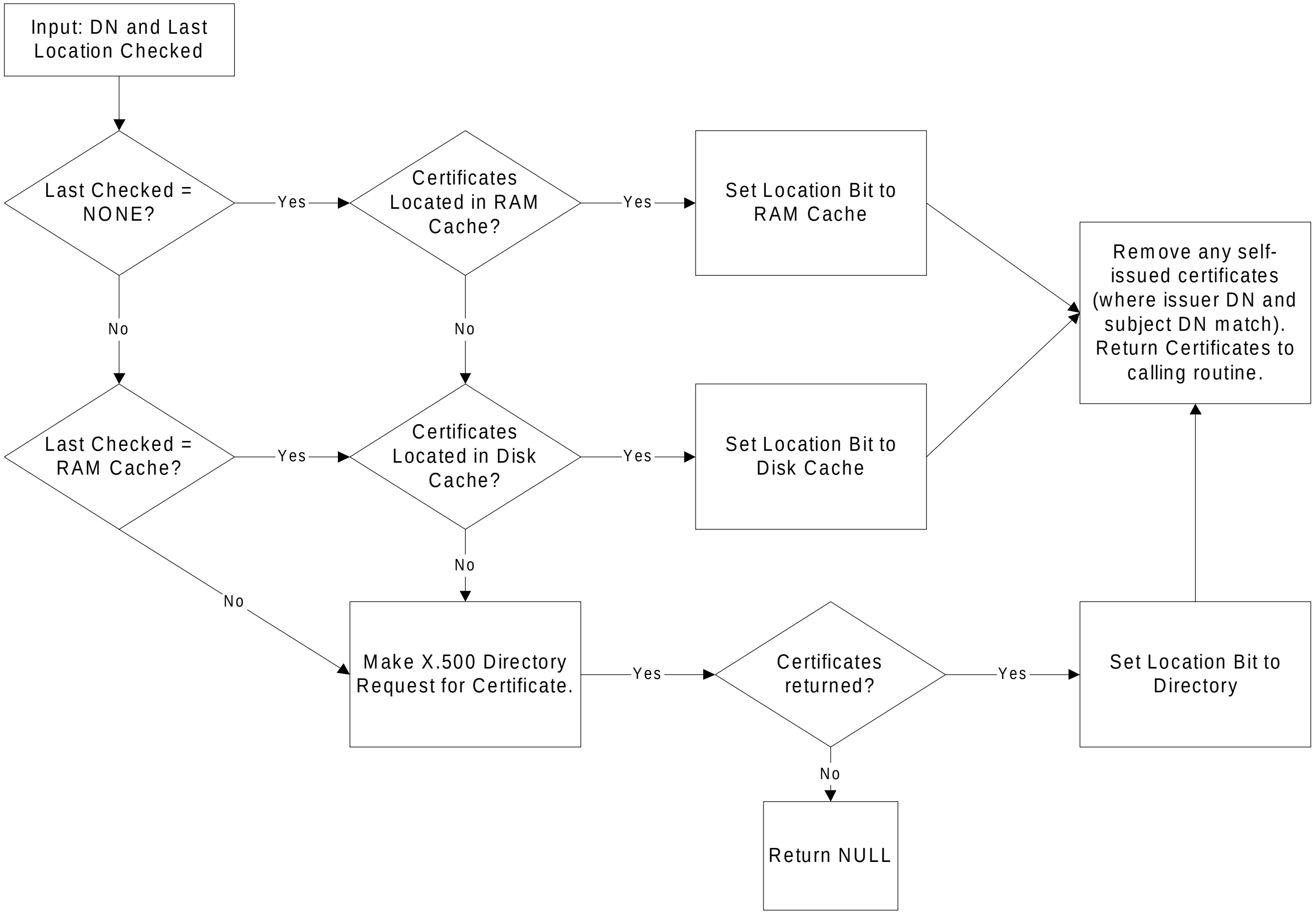
Certificate List Prioritization

- The linked list of certificates should be sorted according to certificate path priorities.
- How should we order these items?
- Are there others?



1) Certificates retrieved from the cACertificate attribute should have priority over certificates retrieved from the crossCertificate attribute
2) Certificates in which issuer algorithm OID = subject algorithm OID should have priority
3) Certificates with longer validity periods (furthest notAfter date) should have priority
4) Certificates that assert policies in the initial-acceptable-policy-set should have priority
5) Certificates with fewer RDN elements in the Issuer DN should have priority
6) Certificates match more rdns between the issuer DN and relying party trust anchor DN should have priority
7) Certificates that match more rdns between the subject DN and the issuer DN should have priority

Retrieving a Certificate



Input: DN and Last Location Checked
Last Checked = NONE?
Yes
Certificates Located in RAM Cache?
Yes
Set Location Bit to RAM Cache
No
No
Last Checked = RAM Cache?
Yes
Certificates Located in Disk Cache?
Yes
Set Location Bit to Disk Cache
Remove any self-issued certificates (where issuer DN and subject DN match). Return Certificates to calling routine.
No
No
Make X.500 Directory Request for Certificate.
Yes
Certificates returned?
Yes
Set Location Bit to Directory
No
Return NULL

CYGNACOM SOLUTIONS